PROVIDING CONFIDENTIALITY IN CONTENT-BASED PUBLISH/SUBSCRIBE SYSTEMS

Mihaela Ion, Giovanni Russello
CREATE-NET International Research Center, via alla Cascata 56/D, Trento, Italy

Bruno Crispo
Department of Information Engineering and Computer Science, University of Trento, Trento, Italy

Keywords: Confidentiality, Publish/subscribe, Attribute-based encryption, Encrypted search.

Abstract: Publish/subscribe is a loosely-coupled communication paradigm which allows applications to interact indirectly and asynchronously. Publisher applications generate events that are sent to interested applications through a network of brokers. Subscriber applications express their interests by specifying filters that brokers can use for routing the events. In many cases it is desirable to protect the confidentiality of events and filters from any unauthorised parties, including the brokers themselves. Supporting confidentiality of messages being exchanged is challenging mainly because of the decoupling of publishers and subscribers who should not have to share keys, and because brokers forward messages based on the actual content of the messages that we desire to keep confidential. This paper argues that a complete solution for confidentiality in pub/sub systems should provide: (i) confidentiality of events and filters; (ii) filters that can express very complex constraints on events even if brokers are not able to access any information on both events and filters; (iii) and finally it does not require publishers and subscribers to share keys. We show that current solutions are not able to provide all these properties at the same time and suggest a possible solution based on attribute-based encryption and encrypted search.

1 INTRODUCTION

The publish/subscribe (pub/sub) model is an asynchronous communication paradigm where senders, known as *publishers*, and receivers, known as *subscribers*, are loosely coupled. The messages that publishers generate are called *events*. Publishers do not send events directly to subscribers, instead a network of interconnected brokers is responsible for events delivery. In fact, publishers do not know who receives their events and subscribers are not aware of the source of information. In order to receive events, subscribers need to register interest with a broker through a *filter*. When a new event is published, brokers forward it to all subscribers which expressed a filter that matches the event.

The pub/sub communication paradigm has the advantage of allowing the full decoupling of the communicating entities (Eugster et al., 2003) which enables dynamic and flexible information exchange between a large number of entities. The communicating parties do not need to know each other or establish contacts in order to exchange content. Moreover, if durable subscription is enabled, publisher and subscribers do not need to actively participate in the interaction at the same time. If a subscriber is offline when a publisher creates an event, the broker will store the event until the subscriber becomes online and the event can be delivered.

Most of the research in pub/sub has been focusing on efficient routing of events. However, there are scenarios that require control over who can access the information. For example, a stock quote service could provide to paying customers information on stock prices. In this case, the events must be delivered only to paying subscribers. At the same time, subscribers may wish to keep the details of their filters private from anybody spying on their interests. Another application scenario is in the medical sector where physicians are notified when certain events happen such as changes in the condition of a patient. Such information should be available to the autho-

Ion M., Russello G. and Crispo B. (2010).
PROVIDING CONFIDENTIALITY IN CONTENT-BASED PUBLISH/SUBSCRIBE SYSTEMS.
In *Proceedings of the International Conference on Security and Cryptography*, pages 287-292
DOI: 10.5220/0002993602870292

rised personnel only to protect patient's privacy.

Providing event and filter confidentiality in pub/sub systems is challenging. First of all, any solution trying to establish shared (group) keys would affect the scalability of the system. Publishers and subscribers are decoupled and should not share keying information. Second, brokers forward messages by matching the content of events against the filters expressed by subscribers. Performing such an operation while keeping the details of events and subscriptions hidden from brokers is not straightforward without affecting the expressiveness of the filter. The filters should at least be able to express disjunctions and conjunctions of equalities and inequalities.

A solution for confidentiality in pub/sub systems should, hence, provide: (i) confidentiality of events and filters, (ii) flexible key management that does not require publishers and subscribers to share keys, and (iii) allow subscribers to express filters that can define any monotonic and possibly non-monotonic (i.e. negation) conditions.

Current solutions for confidentiality in pub/sub systems achieve only partially these goals. For example, in order to support routing based on expressive filters, (Khurana, 2005) and (Raiciu and Rosenblum, 2006) encrypt only certain event fields while other fields are left as cleartext so that they can be used for routing. Other solutions (Raiciu and Rosenblum, 2006) require publishers and subscribers to share a group key which hampers the loosely coupling and scalability of the pub/sub model. (Shikfa et al., 2009) provides confidentiality of events and filters but the filter is restricted to equality with one keyword.

The contributions of this paper are the following. First of all, we provide an analysis of requirements for confidentiality in pub/sub systems. Second, we highlight which are the shortcomings of current solutions that address the confidentiality issue in pub/sub systems. Finally, we sketch a preliminary design of an encryption schema based on existing mechanisms that overcomes the limitations of current solutions.

This paper is structured as follows. Section 2 introduces the pub/sub communication model and motivates the need for confidentiality through an example. Section 3 formally discusses the properties a complete solution for confidentiality should support. Section 4 analyses the most relevant solutions for pub/sub confidentiality based on the properties defined and shows they cannot support all of them at the same time. Section 5 shows how a solution could be achieved by using attribute-based encryption and encrypted search. Section 6 concludes the paper and highlights some future work.

2 THE PUBLISH/SUBSCRIBE COMMUNICATION PARADIGM

Figure 1 shows a pub/sub network connecting several publishers and subscribers. The brokers forward the events created by publishers based on the filters registered by subscribers. Several pub/sub approaches have been proposed which differ in the granularity of the filters. The most simple one is *topic-based*, in which subscribers subscribe to a topic identified by a *keyword* (Zhuang et al., 2001). A topic-based scheme is similar to the notion of group communication. When subscribing to a topic T, a subscriber becomes a member of group T. When an event for topic T is published, the event is broadcasted to all the members of that group. Organizing topics in hierarchies allows a better management of subscriptions (Singhera, 2008). For example, by registering to a topic, a subscriber is also registered to all subtopics.

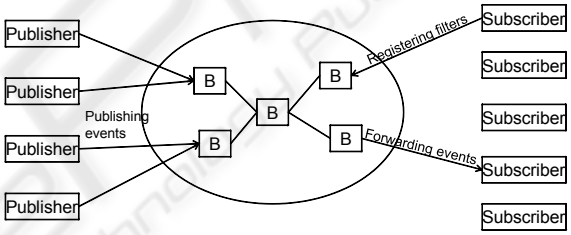

Figure 1: The pub/sub infrastructure connects publishers and subscribers via a network of interconnected brokers.

Topic-based schemes are easy to implement but they offer limited expressiveness. *Content-based* schemes are more flexible and allow expressing subscriptions based on the actual content of the event. To express a filter on the content of an event, subscribers need a query language and understanding of the data formats. For example, in Gryphon (Banavar et al., 1999) and Siena (Carzaniga et al., 2001) the event consist of sets of ($attribute_{name} = attribute_{value}$) pairs and filters are specified as SQL WHERE clauses. Java Message Service (JMS) (Hapner and Stout, 2002) does not allow filtering on the content of the event, but instead, events carry properties in their headers and subscribers can define filters on them. Filters that apply to the composition of simple events have also been proposed ((Bacon et al., 2000)). When expressing such a filter, subscribers are notified upon the occurrence of the composite event.

Because of its generality and expressiveness, we will focus on content-based filtering. We assume that filters define constrains in the form of *name-op-value* where *op* can be one of the comparison operators such as $=, \leq, <, \geq, >$. Constrains can be logically com-

bined using AND, OR and NOT to form complex subscription patterns.

In the following we show an example of an application built using pub/sub in which an attacker could compromise the confidentiality of events and filters.

2.1 A Case for Pub/Sub Confidentiality

Figure 2 shows an example of a Financial News Service implemented using a pub/sub system for information delivery. The publishers can be different stock exchanges and financial news agencies which use the Financial News Service to sell their content to customers. To subscribe to particular content, a customer specifies a filter and contacts the News Service to pay the fee. Then it subscribes with a broker to receive notifications and the broker registers the filter only after the registration is verified with the Financial News Service. When a publisher publishes some new content, the network of brokers will deliver the content to the authorized subscribers. The publisher receives the payment from the Financial News Service without contacting the subscribers directly.

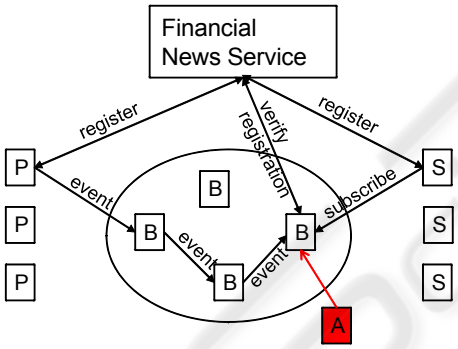

Figure 2: An attacker who is able to corrupt a broker can listen on filters and events.

In a typical pub/sub system where confidentiality is not implemented, an attacker who is able to corrupt a broker could read the traffic that comes in and out the broker. The attacker would be able to read the events without paying the fee and then resell them, and read the filters expressed by the subscribers. To protect from this kind of attacks, it is necessary to protect the content of notifications and filters.

3 CONFIDENTIALITY IN PUB/SUB SYSTEMS

In this section we formalize the confidentiality requirements for the information exchanged in pub/sub systems. We start by defining the threat model. We then describe general goals identified in literature from which we derive our more specific requirements.

3.1 Attack Model

We assume an honest-but-curious model for publishers, brokers and subscribers, as in (Srivatsa and Liu, 2007; Shikfa et al., 2009). This means that the entities follow the protocol, but may be curious to find out information by analysing the messages that are exchanged. For example, a broker may try to read the content of an event or try to learn the filtering constrains of subscribers. Subscribers may want to read the events delivered to other subscribers. We also assume that a passive attacker outside the pub/sub system may be able to listen on the communication and invade the privacy of the participants.

3.2 Confidentiality Goals

The problem of confidentiality in content-based pub/sub systems has first been analysed in (Wang et al., 2002) where the authors proposed general goals without giving a concrete solution. Three issues where identified:

- Publication confidentiality: only authorised publishers should be able to access the events.
- Subscription confidentiality: the details of the filters are hidden from publishers and brokers (or other unauthorised parties).
- Information confidentiality: the brokers should be able to perform content-based routing without learning any information about the event or filter.

Providing these goals is still an open issue due to the following challenges. First of all, a basic encryption scheme would require publisher and subscribers to share a secret key. This is not desirable in a pub/sub system because it would weaken the decoupling property. Second, brokers would need to execute matching operations on encrypted events and filters which is not simple using basic techniques.

3.3 Confidentiality Requirements

The goals introduced above are not specific enough and do not fully reflect the particularities of the pub/sub systems. We formalize these goals into more concrete requirements to be implemented by an encryption and routing scheme for pub/sub systems:

- (P1) confidentiality of events;
- (P2) confidentiality of filters;

- (P3) a simplified and scalable key management that does not require publishers and subscribers to share keys, hence fully supporting the loosely-coupled model of the pub/sub paradigm;
- (P4) allowing brokers to execute matching of encrypted events against complex encrypted filters. By complex encrypted filters we mean filters that can express conjunctions and disjunctions of equalities, inequalities and negations in an encrypted form.

(Shikfa et al., 2009) showed that both event (P1) and filter (P2) confidentiality are required to effectively reduce the risk of leaking information in a pub/sub system. For instance, in providing only subscription confidentiality an attacker who knows the content of the event may infer the subscription filter. We argue that (P3) and (P4) should also be provided in order to have a complete solution for confidentiality.

In the following section, we analyse current solutions and show they cannot support in the same time all the above properties.

4 RELATED WORK

(Khurana, 2005) proposes a scheme that targets confidentiality of events but not of filters. Events are encoded in XML format and only specific fields (e.g., price) are encrypted with a symmetric key k. The publisher then encrypts k with its public key and attaches it to the message. The brokers forward the events based on the fields left unencrypted and a proxy service changes the encryption of k to an encryption with the public key of the subscriber. This solutions achieves partially (P1) encrypting only specific fields but not the entire event. Properties (P2) and (P4) are not addressed since events are forwarded based on unencrypted event fields and filters. Key management is scalable and does not require publishers and subscribers to share a key, hence achieving (P3).

(Raiciu and Rosenblum, 2006) target simultaneous event and filter confidentiality (P1 and P2). In their model, notifications are composed of $(name, value)$ pairs where only value is encrypted (P1 is thus only partially achieved). Publishers and subscribers are required to share a group key (P3 is not achieved) which is used to encrypt events and filters. The method does not support general filters as defined in (P4), but is limited to equality filtering, some range matches and keyword matching. To hide the name of the attribute, it is possible to concatenate it with the attribute type and size and then hash them.

(Srivatsa and Liu, 2007) propose a specific hierarchical key management scheme that achieves confidentiality of events (P1) and filters (P2). A trusted centralized authority distributes encryption keys to publishers and authorization keys to subscribers. To support range matching, keys are organized in a hierarchical structure, each key corresponding to an interval. An authorization key corresponds to a filter and is able to derive the encryption key for an event that matches the filter. The key management scheme does not require publishers and subscriber to share keys (achieving P3). Each event has a routable topic attribute which is encrypted using an encrypted search technique. To prevent dictionary attacks on the events, the routable attributes are tokenized and transformed into pseudo-random chains. The approach is vulnerable to inference attacks which use information about the frequency at which events are published to learn information about an event. To prevent these attacks, a probabilistic multi-path event routing scheme is proposed at the cost of extra overhead. The main disadvantage of this method is that it supports routing based on only one keyword (the topic), hence not achieving P4). It is possible to express inequality conditions but they can only be checked at the subscriber side and not by the brokers. When the subscriber receives an event matching the expressed topic, the authorization key of the subscriber will allow deriving a correct decryption key only if the numerical value of the attribute is in the range specified by the subscriber.

In (Shikfa et al., 2009), Shikfa et al. propose a solution based on multiple layer commutative encryption that achieves content and filter confidentiality (P1 and P2). The method has the advantage that it does not require publishers and subscribers to share keys, instead it uses a local key management in which each node needs to share a secret key with the immediate r neighbours (P3 is achieved). To avoid collusion attacks, r can be set as big as necessary. If r consecutive nodes collude, they can decrypt their children's subscriptions, but not the subscriptions of other nodes. The main drawback is that events and filters contain only one keyword. Encrypted routing tables are created for a single keyword and the matching is basically an equality test (P4 is not achieved).

Concluding the related work, we observe that in order to provide confidentiality of events and filters, current solutions limit the expressiveness of the filter. Where more complex filters are allowed, confidentiality is provided only for specific attributes. Our goal is to propose a solution that can achieve both at the same time while keeping the key management simple and scalable.

5 SOLUTION IDEA

In this section we sketch a preliminary idea for a solution that satisfies all the properties we identified. In the following we assume that an event E consists of: (i) the message M that represents the content of the event and (ii) a set of attributes $\{a_i\}$ that characterise M and are used for event filtering by the brokers. A filter consists of conjunctions and disjunctions of equalities, inequalities and negations.

5.1 Confidentiality of Events

Confidentiality of events (P1) can be achieved by means of encryption. We require a mechanism that allows authorised subscribers to decrypt events without establishing shared keys with the publishers (e.g., group keys). From the available encryption schemes, ciphertext policy attribute-based encryption (CP-ABE) (Bethencourt et al., 2007) would allow publishers to encrypt the content of the event by specifying the characteristics that subscribes must satisfy to obtain the cleartext of the event. Indeed, in CP-ABE policies (or access structures) are associated with ciphertexts and attributes are associated with keys. This is similar to the capability model in access control. A key can decrypt a ciphertext if its associated attributes satisfy the policy associated with the data. Using CP-ABE, a publisher could specify that only paying subscribers registered with a particular system can decrypt the message M. In this way, we are effectively decoupling the encryption of events at the publisher site from its decryption at the subscriber site, thus simplifying the key management process (P3).

5.2 Filter Confidentiality and Secure Filtering

Filter confidentiality (P2) can also be achieved by means of encryption. In key-policy attribute-based encryption (KP-ABE) (Goyal et al., 2006) ciphertexts are labelled with sets of attributes and private keys are associated with access structures. A key is able to decrypt a ciphertext if its associated access structure is satisfied by the attributes of the ciphertext. The access structure, represented as a tree, allows expressing any monotone access formula consisting of AND, OR, or threshold gates, inequalities (Bethencourt et al., 2007), and negations (i.e., the attribute must not be present among the attributes of the ciphertext) (Ostrovsky et al., 2007). Expressing the filters as access structures using KP-ABE would allow us to satisfy property (P4). However, KP-ABE does not encrypt the access tree. We require an additional mechanism that allows subscribers to encrypt the attributes of the filters in such a way that brokers are able to decide whether the attributes from the tree match the attributes in the set $\{a_i\}$ attached by the publisher to the event (also encrypted).

Multi-user searchable encrypted data (SDE) (Dong et al., 2008) allows an untrusted server to perform encrypted searches on data without revealing the data or the keywords to the server. The advantage of this method is that it allows multi-user access without the need for a shared key between users. Each user in the system has a unique set of keys. The data encrypted by one user can be decrypted by any other authorised user. The scheme is built on top of proxy encryption schemes. The idea is that a user defines a set of keywords for each document. The keywords and document are encrypted using proxy encryption and stored on the server. When a user wants to search for a document, it needs to create a trapdoor for each keyword. The trapdoor is used by the server to match the search keywords against the keywords of the stored document. The server can identify a match without learning the keyword.

To achieve (P4) we combine KP-ABE with SDE. In particular, a subscriber S defines a filter F as KP-ABE access tree and encrypts the attributes of the filter using SDE. The publisher also encrypts the set set of attributes $\{a_i\}$ using SDE. When the event E reaches a broker, if the set of attributes associated with the event satisfy the filter F, the broker knows that the event can be forward to S without learning anything about the attributes or filter. In KP-ABE, when the attributes of the ciphertext match the access tree of the key, the holder of the key is able to decrypt the ciphertext. However, by first encrypting M using CP-ABE, the broker does not gain any information on the actual content of the event.

In this section we have shown how to combine CP-ABE, KP-ABE and SDE to achieve event and filter confidentiality, complex filters that can express any access structure, while not requiring publishers and subscribers to share key. In the following we show how to apply this to the Financial News Service example.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we describe requirements for providing confidentiality in pub/sub systems. We show that the identified requirements are not supported by any

available solutions and propose a solution which can achieve all based on CP-ABE, KP-ABE and multi-user SDE. Our goal is to support both publication and subscription confidentiality while not requiring publishers and subscribers to share secret keys. Although events and filters are encrypted, brokers should perform event filtering without learning any information. Finally, subscribers should be able to express filters that can define any monotonic and non-monotonic constraints on events.

As future work we are planing to provide a concrete design of our solution. We will also provide a formal security analysis of our scheme. We will implement the schema and integrate it with a mainstream pub/sub system in order to evaluate the introduced overhead.

ACKNOWLEDGEMENTS

The work of the third author is partially funded by the EU project MASTER contract no. FP7-216917.

REFERENCES

Bacon, J., Moody, K., Bates, J., Hayton, R., Ma, C., McNeil, A., Seidel, O., and Spiteri, M. (2000). Generic support for distributed applications. *IEEE Computer*, 33(3):68–76.

Banavar, G., Chandra, T., Mukherjee, B., Nagarajarao, J., Strom, R., and Sturman, D. (1999). An efficient multicast protocol for content-based publish-subscribe systems. In *International Conference on Distributed Computing Systems*, volume 19, pages 262–272. IEEE COMPUTER SOCIETY PRESS.

Bethencourt, J., Sahai, A., and Waters, B. (2007). Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334. Citeseer.

Carzaniga, A., Rosenblum, D., and Wolf, A. (2001). Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems (TOCS)*, 19(3):332–383.

Dong, C., Russello, G., and Dulay, N. (2008). Shared and Searchable Encrypted Data for Untrusted Servers. *Lecture Notes in Computer Science*, 5094:127–143.

Eugster, P., Felber, P., Guerraoui, R., and Kermarrec, A. (2003). The many faces of publish/subscribe. *ACM Computing Surveys (CSUR)*, 35(2):131.

Goyal, V., Pandey, O., Sahai, A., and Waters, B. (2006). Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, page 98. ACM.

Hapner, M., B. R. S. R. F. J. and Stout, K. (2002). Java message service. Sun Microsystems Inc., Santa Clara, CA.

Khurana, H. (2005). Scalable security and accounting services for content-based publish/subscribe systems. In *Proceedings of the 2005 ACM symposium on Applied computing*, page 807. ACM.

Ostrovsky, R., Sahai, A., and Waters, B. (2007). Attribute-based encryption with non-monotonic access structures. In *Proceedings of the 14th ACM conference on Computer and communications security*, page 203. ACM.

Raiciu, C. and Rosenblum, D. (2006). Enabling confidentiality in content-based publish/subscribe infrastructures. *Securecomm and Workshops*, 28:1–11.

Shikfa, A., Onen, M., and Molva, R. (2009). Privacy-Preserving Content-Based Publish/Subscribe Networks. In *Emerging Challenges for Security, Privacy and Trust: 24th Ifip Tc 11 International Information Security Conference, SEC 2009, Pafos, Cyprus, May 18-20, 2009, Proceedings*, page 270. Springer.

Singhera, Z. (2008). A workload model for topic-based publish/subscribe systems.

Srivatsa, M. and Liu, L. (2007). Secure event dissemination in publish-subscribe networks. In *Proceedings of the 27th International Conference on Distributed Computing Systems*, page 22. Citeseer.

Wang, C., Carzaniga, A., Evans, D., and Wolf, A. (2002). Security issues and requirements for Internet-scale publish-subscribe systems. In *PROCEEDINGS OF THE ANNUAL HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES*, pages 303–303.

Zhuang, S., Zhao, B., Joseph, A., Katz, R., and Kubiatowicz, J. (2001). Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video*, page 20. ACM.